International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 12 Issue: 04 | Apr 2025 www.irjet.net p-ISSN: 2395-0072

DISPLAY ON FPGA WITH 7 SEGMENT CONTROLLER

J.Yasaswi[1], Visranthamma [2], N. Chandana ravithreni [3], V.Lakshmaiah [4] , V.LURDHU MARREDDY[5]

[1]Student & AMRITA SAI INSTITUTE OF SCIENCE AND TECHNOLOGY
[2]Assistant Professor & AMRITA SAI INSTITUTE OF SCIENCE AND TECHNOLOGY
[3]Student & AMRITA SAI INSTITUTE OF SCIENCE AND TECHNOLOGY
[4]Student & AMRITA SAI INSTITUTE OF SCIENCE AND TECHNOLOGY
[5]Student & AMRITA SAI INSTITUTE OF SCIENCE AND TECHNOLOGY

Abstract -

In modern digital systems, the use of Field-Programmable Gate Arrays (FPGAs) has gained significant traction due to their reconfigurability, speed, and adaptability for diverse applications. One of the most fundamental tasks in FPGA-based systems is the implementation of displays, such as seven-segment displays. Seven-segment displays are widely used for visualizing numerical data, commonly seen in digital clocks, counters, and various consumer electronics. The design of an FPGA-based seven-segment controller offers a versatile, efficient, and customizable solution to display a wide array of numeric and alphanumeric values.

This research paper presents the design and implementation of a seven-segment display controller using FPGA technology. The proposed system involves the integration of a Binary Coded Decimal (BCD) to seven-segment converter, a slow clock generator, and a scrolling message display mechanism to create a dynamic display output. The paper demonstrates the usage of VHDL and Verilog hardware description languages to describe the modules that work together to generate the desired display output.

The system is composed of several key components, each playing a crucial role in ensuring proper operation and functionality. The first major component is the BCD to 7-segment converter, which takes a 4-bit binary input (BCD) and converts it into the appropriate seven-segment display pattern. This conversion is essential for rendering numbers on the display. The second component is a slow clock generator that produces clock signals with slower frequencies to control the timing of the display update. The system includes both a 1 Hz slow clock and a 100 Hz slow clock, which are necessary for controlling the rate at which data is updated on the seven-segment displays. The third component is the scrolling message module, which orchestrates the sequencing of multiple numbers or characters across four different seven-segment displays.

Keywords : *FPGA, 7 segment Controller,Clock , Frequency*

1.INTRODUCTION

In today's fast-paced technological world, embedded systems have become an integral part of various industries. These systems are responsible for controlling a wide range of devices and operations, from home appliances and industrial machines to automotive systems and telecommunications. Among the many types of embedded systems, digital display systems have seen widespread usage, particularly those that utilize seven-segment displays. Seven-segment displays are a simple yet powerful solution for displaying numerical and alphanumeric data, commonly seen in devices like digital clocks, counters, calculators, and other digital instruments. The utility of these displays lies in their ability to represent numerical data in an easily readable and visually appealing format, making them indispensable in both everyday applications and specialized industrial uses.

The technology behind controlling and driving a seven-segment display has evolved over the years, with significant advancements made in terms of flexibility, processing power, and adaptability. In recent years, Field-Programmable Gate Arrays (FPGAs) have emerged as a popular platform for digital system design due to their reconfigurability, speed, and parallel processing capabilities. An FPGA is an integrated circuit that can be programmed to perform specific tasks, such as controlling displays, processing signals, and executing complex algorithms. Unlike traditional microcontrollers, FPGAs offer a unique advantage in handling high-speed operations, performing parallel computations, and providing customizability, making them an excellent choice for digital display applications.

This research paper presents the design and implementation of a seven-segment display controller using an FPGA. The system utilizes a combination of Verilog hardware description language (HDL) modules that work together to generate dynamic, real-time messages on the display. The core functionality of the system lies in the ability to control and manipulate the state of multiple seven-segment displays to visualize data, which is a fundamental task in many embedded systems applications. The key components of the system include a

Binary Coded Decimal (BCD) to seven-segment converter, slow clock generators, and a scrolling message display mechanism. Each of these components is designed to function independently and cooperatively to ensure smooth, efficient, and error-free operation of the entire display system.

The primary objective of this paper is to provide a comprehensive overview of the design process for the seven-segment display controller, focusing on the integration of these components into an FPGA. The system is designed to display a scrolling message, where characters are sequentially shifted across a series of four seven-segment displays. This functionality is achieved through a series of timed clock signals, which are generated using slow clock modules operating at different frequencies. These slow clocks allow for the precise control of when and how each segment of the display is activated, providing a smooth scrolling effect.

The first essential module in the design is the BCD to seven-segment converter. This module is responsible for taking a 4-bit Binary Coded Decimal (BCD) input and converting it into the appropriate signals required to drive the seven-segment display. BCD is a binary representation of decimal numbers, where each 4-bit group represents a decimal digit. The converter outputs a 7-bit signal that corresponds to one of the seven segments in the display. For example, if the input is 0, the converter will output a signal that turns on the appropriate segments to display the digit '0'. This module is crucial for ensuring that the correct digit is displayed based on the BCD input.

The second important component is the slow clock generator. Since the seven-segment displays need to be updated at a relatively low rate (for example, once per second), the clock signals used to drive the display need to be slowed down. Two clock generators are used in the system: one operates at a frequency of 1 Hz, and the other at 100 Hz. The 1 Hz clock is used to control the overall scrolling speed, ensuring that the characters on the display change smoothly over time. The 100 Hz clock is used to manage the switching between different segments of the display, ensuring that each display is updated at the correct interval. The slow clocks are essential for creating a smooth scrolling effect and preventing the display from flickering.

The third key component is the scrolling message module, which is responsible for shifting the characters across the display. This module controls which digits are displayed on each of the four segments of the display. It uses a shift register mechanism to move the digits from one display to the next, creating the effect of scrolling text. The scrolling message is achieved by updating the content of the display at regular intervals, using the slow clock signals to synchronize the changes. The scrolling message module also incorporates a reset feature, allowing the display to start over from the beginning if needed.

The integration of these components into a single FPGA design creates a flexible, efficient, and reliable system for controlling seven-segment displays. One of the key advantages of using FPGA technology in this project is its ability to handle multiple tasks in parallel. FPGAs can execute several operations simultaneously, allowing for more efficient processing of the slow clocks, display updates, and message scrolling. This parallelism ensures that the system operates smoothly and with minimal delay, even when multiple displays are being controlled at the same time.

Moreover, FPGAs offer the flexibility to modify and customize the design based on specific requirements. The modular nature of the system allows for easy updates and enhancements, such as adding more displays, implementing different scrolling effects, or integrating the system with external devices. The reconfigurability of FPGAs ensures that the design can be adapted to a wide range of applications, from simple numeric displays to more complex alphanumeric outputs.

Another benefit of using an FPGA for this design is the improved performance over traditional microcontroller-based systems. Microcontrollers typically execute instructions sequentially, while FPGAs can process multiple signals and operations in parallel, leading to faster response times and more efficient use of resources. In the case of the seven-segment display controller, the FPGA's parallel processing capabilities allow for more precise control over the display, minimizing delays and ensuring that the scrolling message appears smooth and continuous.

Additionally, the FPGA-based design offers scalability. As the need for more complex displays grows, the system can easily be expanded by adding more display modules or modifying the clock frequencies. The modularity and reconfigurability of FPGA designs make them ideal for handling more advanced tasks, such as controlling larger display arrays, integrating sensors, or interacting with other systems via communication protocols like I2C or SPI.

The significance of this work extends beyond the design of a simple display controller. It serves as a foundational study for anyone interested in learning about FPGA-based display systems, particularly in the context of digital display control. The techniques and methods presented here can be applied to more advanced systems, such as real-time clocks, digital signage, and interactive user interfaces, and can be adapted for use in a wide range of applications.

2. HARDWARE REQUIREMENTS

The hardware requirements for implementing a seven-segment display controller using an FPGA are relatively straightforward, but they must be carefully selected to ensure the system functions efficiently and meets the desired specifications. At the core of the system lies the FPGA, which is the primary component for controlling the logic and timing required to drive the display and handle the scrolling message functionality. The FPGA must have sufficient logic resources, such as Look-Up Tables (LUTs), flip-flops, and routing resources, to accommodate the design, which includes several modules like the BCD to seven-segment converter, clock generators, and the scrolling display logic.

For the FPGA implementation, a development board with an FPGA chip that includes I/O pins for driving the seven-segment displays is necessary. Common FPGA platforms such as the Xilinx Spartan-6 or Altera Cyclone IV can be used, as they offer a good balance between performance and cost for relatively simple display-based projects. The FPGA must also have enough I/O pins to interface with all the necessary components. In this case, at least 4 pins are required for controlling four seven-segment displays. These pins will be used for the segment control (a to g), while additional pins may be needed for the common anode or common cathode configurations, depending on the type of seven-segment display used.

The seven-segment displays themselves are essential components in this project. These displays consist of seven individual LEDs arranged in a figure-eight shape, which can be illuminated to represent digits or letters. In this system, the displays will be driven using a common cathode or common anode configuration, depending on the specific type chosen. The common cathode configuration is more commonly used, where each segment is controlled by sending a high or low signal to it. The displays will be connected to the FPGA via the I/O pins, and each segment will need to be connected to an individual output of the FPGA.

Additionally, for controlling the scrolling message and generating the clock signals required for the display updates, clock generators must be integrated into the system. For this, two slow clock modules are needed: one generating a 1 Hz clock for the scrolling effect and another generating a 100 Hz clock to manage the multiplexing of the displays. These clocks can be generated using external oscillators or directly within the FPGA using an internal clock source. The FPGA must be able to generate these clock signals reliably and distribute them to the appropriate parts of the system.

A reset button or switch is another important hardware component to facilitate system initialization. The reset button will be used to ensure that the system starts from a known state, allowing the scrolling message to begin from the start when needed. This is a basic but essential feature for ensuring proper operation during power-up or during manual resets.

For power supply, the FPGA board and the seven-segment displays require a stable voltage. Typically, the FPGA operates at 3.3V or 5V, depending on the specific FPGA model used, and the seven-segment displays typically require 5V. Therefore, a stable 5V power supply is necessary to drive both the FPGA and the displays. It is important to ensure that the power supply is capable of providing enough current for both the FPGA and the displays, especially when multiple segments are lit simultaneously.

In addition, jumper wires, breadboards, or PCB (Printed Circuit Board) will be necessary for making the connections between the FPGA and the seven-segment displays, as well as for connecting any other components like the clock generator or reset switch. If the project is to be extended in the future to accommodate more displays or more advanced features, the design should take into account the scalability of the hardware setup.

Finally, programming hardware and software tools, such as a JTAG programmer, will be required to load the design onto the FPGA. The JTAG programmer connects to the FPGA via a dedicated programming interface and allows the user to upload the Verilog code to the FPGA. Additionally, a PC or laptop will be required for writing and compiling the Verilog code, which will then be uploaded to the FPGA for execution.

In summary, the key hardware requirements for this project include an FPGA development board, seven-segment displays (common cathode or anode), clock generators, power supply components, a reset mechanism, and necessary connecting materials such as jumper wires, breadboards, or PCBs. These components work together to form the foundation of the seven-segment display controller system, ensuring proper function and flexibility for future enhancements or adaptations.

3. Implementation

The implementation of a seven-segment display controller with FPGA involves creating a system that efficiently manages the control and multiplexing of multiple seven-segment displays, while also generating the necessary clock signals for updating and scrolling the displayed information. The goal is to design a solution that can handle a simple display system, with the ability to show a scrolling message, and utilize FPGA's capability to manage time-based and display-specific tasks effectively. This is accomplished through a combination of Verilog modules that handle different aspects of the design.

The first key module in this design is the BCDto7Seg module, which is responsible for converting a 4-bit BCD (Binary Coded Decimal) input into a corresponding 7-bit output that controls the segments of a seven-segment display. Each of the digits 0 to 9 has a unique 7-bit pattern, which is crucial for displaying numerical information or characters on the display. In this case, the BCD input could represent numbers or even simple characters, such as 'hi' or a dash, based on the desired functionality. This conversion ensures that the system can accurately control each seven-segment display segment, lighting up the appropriate parts to form readable digits.

Next, clock generation is a critical part of this system, as it ensures proper timing for both scrolling and multiplexing of the seven-segment displays. Two separate clock generator modules are implemented: slow_clock_1hz and slow_clock_100hz. The slow_clock_1hz module generates a 1 Hz clock, which is used to control the scrolling behavior of the message across the displays. This clock ensures that the message scrolls in a controlled and visible manner by updating the displayed digits at a 1-second interval. On the other hand, the slow_clock_100hz module generates a 100 Hz clock, which is used for the multiplexing of the displays. This ensures that each of the four displays is updated in quick succession, giving the illusion of simultaneous display updates while actually switching between displays very rapidly.

The scrolling message functionality is implemented within the scrolling_message module. This module controls the sequence of digits that are displayed on the seven-segment displays, shifting the displayed message across the four displays. The scrolling is triggered by the 1 Hz clock, which is used to determine when to shift the message. Each cycle of the 1 Hz clock results in a new set of digits being loaded into the display registers, creating the effect of a scrolling message. The logic inside this module handles the cycling of values for the four displays, setting which digit appears on which display at any given moment. Additionally, the reset functionality is integrated into this module, allowing for the system to be restarted and initialized to its starting state when required.

The multiplexing process ensures that only one display is active at any given time, which is crucial to preventing all displays from lighting up simultaneously and causing confusion. By rapidly cycling between the displays at 100 Hz, the system makes use of the human eye's persistence of vision, allowing all four displays to appear as if they are being updated simultaneously, even though they are updated one at a time. The bcd7seg module, responsible for converting BCD inputs into 7-segment display outputs, is invoked in the scrolling message module to ensure that the correct display patterns are shown on each of the four seven-segment displays.

Once all the individual modules are developed, the next step is integrating them into the main FPGA system. This involves connecting the various modules together, ensuring that they communicate correctly and that the display patterns are generated in sync with the clock signals. The output of the bcd7seg module is connected to the segments of the seven-segment display, and the outputs of the clock generation modules are fed into the scrolling message module to drive the timing for both scrolling and multiplexing. Additionally, the FPGA I/O pins are connected to the displays in such a way that they control the segment and common anode or cathode lines, which determine which segments light up.

The final stage of the implementation involves synthesizing the design and loading the bitstream into the FPGA. Once the bitstream is loaded, the system can be tested on the FPGA hardware. This involves observing the scrolling behavior of the message across the seven-segment displays, checking that each digit updates at the correct time and that the multiplexing occurs smoothly. If any issues are detected, such as improper timing or incorrect display patterns, the design is modified, and the bitstream is reloaded into the FPGA for further testing.

3.1 Hardware Integration

The hardware integration of the seven-segment display controller with FPGA involves connecting the essential components such as the FPGA development board, seven-segment displays, and necessary input/output connections to ensure proper functionality. The core hardware used for this project includes an FPGA development board, a set of four common cathode seven-segment displays, and associated components like resistors and wiring for signal routing.

The FPGA development board is the central processing unit of this project. It handles the logic required to convert BCD inputs into seven-segment patterns, manage clock signals, and control the multiplexing of the displays. The FPGA is programmed to execute the Verilog code that implements the BCD to seven-segment decoder, clock generation, and multiplexing functionality. The development board must have enough I/O pins to accommodate the signals necessary for controlling four seven-segment displays. Typically, this involves at least 28 I/O pins (7 segments per display, with 4 displays in total), but the exact number will depend on the specific FPGA model being used.

Each seven-segment display is connected to the FPGA through a series of I/O pins. Since these displays are typically common cathode, each of the seven segments of the display (labeled a through g) is connected to a separate FPGA I/O pin. The common cathode pin of each display is connected to a ground reference, while each segment is

driven high or low to control whether it lights up or stays off. To ensure proper operation, current-limiting resistors are placed in series with each segment to prevent excessive current draw, which could damage the FPGA or the display. These resistors also help to optimize the brightness of the segments.

Additionally, the FPGA development board includes clock inputs, which are used to drive the clock generation modules that control the timing of the scrolling message and multiplexing behavior. These clock signals are essential for timing the transitions between the different digits displayed on the seven-segment displays. The FPGA's I/O pins are connected to the inputs of the clock generation modules to ensure accurate synchronization between the scrolling, display updates, and multiplexing.

To drive the message display, the FPGA interacts with the seven-segment displays by cycling through them at a rate that is fast enough to give the appearance of simultaneous display updates. The FPGA's I/O pins also handle the multiplexing of the displays, ensuring that only one display is active at a time, even though all displays are updated in rapid succession. This multiplexing is achieved by sequentially activating the common cathode pin of each display, while the corresponding segment values are provided to the seven segment connections.

The clock signals, both slow (1 Hz) and faster (100 Hz), are routed through the FPGA and used to control the timing of the display updates. The slow clock signal governs the scrolling of the message, updating which digits are shown on the displays, while the 100 Hz clock ensures proper multiplexing of the displays.

In terms of power supply, the FPGA development board is powered via the provided USB or external power source, which also powers the connected displays. Since seven-segment displays may consume higher currents, proper power management is critical. Depending on the FPGA board's voltage specifications and the display requirements, a voltage regulator might be used to ensure stable and adequate power supply to both the FPGA and the displays. Proper grounding is also crucial to avoid signal interference or malfunctioning of the system.

Finally, physical connection and wiring are key aspects of the hardware integration. The FPGA I/O pins are carefully routed to the correct segments and common cathode pins of the displays, ensuring that each pin performs its intended function. Properly laying out these connections on a breadboard or PCB is essential to avoid short circuits, signal interference, or errors in display behavior.

In summary, hardware integration in this project involves the careful connection of an FPGA development board with multiple seven-segment displays, appropriate resistors for current limiting, clock signal management, and proper power supply to ensure reliable operation. The FPGA handles the logic for decoding, timing, and multiplexing, while the displays are driven by the output signals, creating a smooth and dynamic scrolling message.

3.2 Software Development

Software development for the seven-segment display controller on FPGA involves writing the necessary Verilog code to implement various functions, including BCD to seven-segment conversion, clock signal generation, and multiplexing of multiple seven-segment displays to create a seamless visual experience. The core tasks of software development include designing the logic to control the display of numerical and alphanumeric characters, generating the necessary clock signals, and multiplexing the displays to ensure that the scrolling message is displayed correctly.

The first step in software development is creating the BCD to seven-segment decoder. This module converts binary coded decimal (BCD) input into the corresponding seven-segment display patterns. The Verilog code for this part defines a bcd7seg module that accepts a 4-bit input (y) representing the BCD number and generates the 7-bit output (disp) that drives the seven-segment display. The case statement inside the module maps each BCD value (0-9) to a specific 7-bit pattern that corresponds to the appropriate segments of the display. For example, a BCD value of 0 might result in the display pattern 1111110, which lights up the segments required to display the digit 0.

The second key component is the clock generation. Since FPGA operates at a much faster clock speed than the human eye can detect, a clock division technique is used to slow down the FPGA's clock signal to a manageable speed. The project uses two clock generation modules: slow_clock_1hz and slow_clock_100hz. The slow_clock_1hz module takes the input clock (clk_in) and divides it down to a 1 Hz clock signal, while the slow_clock_100hz module generates a 100 Hz clock. These slow clock signals are crucial for controlling the timing of various processes in the system, such as scrolling the message and updating the display in a multiplexed fashion. The clock generation modules rely on counters that increment on each rising edge of the input clock and reset once they reach the required value. This division ensures that the clock signals operate at the desired frequencies for proper timing control.

The third essential part of the software is multiplexing. To drive multiple seven-segment displays using the limited number of available I/O pins on the FPGA, the displays are multiplexed, meaning that only one display

is active at any given time, but it changes quickly enough to create the illusion of all displays being active simultaneously. The scrolling_message module handles the multiplexing. It uses the 1 Hz clock signal to scroll through four 4-bit digits (representing the characters to be displayed) and updates the active display every 100 ms using the 100 Hz clock. The scrolling_message module also contains a count variable, which cycles through the four digits in a manner that ensures each display is updated in turn. By activating the common cathode pins of each display in sequence and driving the appropriate segment values, the system achieves effective multiplexing.

Additionally, the scrolling_message module contains logic to control the scrolling of the message. The 4-bit digits are shifted across the display sequentially, which makes it appear that the message is moving from one side of the display to the other. The scroll register keeps track of the position of the message, and based on the current value of scroll, the correct 4-bit values for the digits are selected. For example, in one cycle, the first digit might be displayed on the first seven-segment display, and as the message scrolls, the values for the second, third, and fourth digits are updated accordingly.

The an signals (active-low anodes) control which of the four displays is currently active, and they are updated within the scrolling_message module. By enabling only one display at a time and rapidly cycling through all four, the user perceives the message being displayed on all four digits simultaneously. The bcd7seg module is called to decode the 4-bit digit (d) to its corresponding seven-segment pattern, which is then displayed on the active segment of the seven-segment display.

To ensure correct timing and synchronization of the message scrolling and display updates, the software is carefully written to use the generated clock signals. The multiplexing operation happens at a rate fast enough (100 Hz) to ensure that the human eye perceives a continuous display without flicker. The 1 Hz clock signal is used to control the movement of the message, ensuring that the digits shift every second, creating the scrolling effect.

The final software functionality includes the management of a reset signal, which is used to clear the display and restart the scrolling message. This ensures that the system can be easily reset to its initial state, allowing for new messages or system re-initialization without manual intervention.

In summary, the software development for this project involves writing Verilog code to handle the BCD to seven-segment decoding, clock generation, and multiplexing of the displays. The use of two clock signals (1 Hz and 100 Hz) enables smooth scrolling and multiplexing of four seven-segment displays. The combination of these components allows for the creation of a dynamic, scrolling message on the FPGA, with the display updated at regular intervals to ensure smooth visual effects.

4. Real Time Implementation

Real-time implementation of a seven-segment display controller on an FPGA involves translating the design from simulation into a physical system that can interact with the real world. The primary goal is to create a working system that continuously updates and displays information on a set of seven-segment displays in real-time. This process combines both hardware and software components, ensuring that the FPGA, using programmed logic, can control the display, handle inputs, and manage timing in a synchronized manner. The real-time operation of the system requires precise coordination between various elements of the hardware, including clock generation, display multiplexing, and the logic for managing the display of data.

To begin with, the design of the system architecture is crucial in ensuring that the FPGA will be able to handle all the required tasks. The FPGA will serve as the primary controller, decoding input binary values (in BCD format) and translating them to the appropriate seven-segment display pattern. It will also need to generate timing signals that control the display refresh rate and scrolling behavior, ensuring that the displayed information is updated at a human-readable speed. The system is set up in such a way that the FPGA handles the logic for multiplexing the four seven-segment displays. By rapidly switching between each display, the FPGA can give the illusion that all four displays are active simultaneously, which is essential for displaying data in real-time.

The hardware setup for the real-time implementation is designed to ensure that all components function seamlessly. The FPGA development board, such as the Xilinx Spartan or Altera Cyclone, is chosen for its ability to handle the programmable logic required for the task. The seven-segment displays, which are typically of the common cathode variety, are connected to the FPGA. These displays consist of seven LED segments, each of which can be illuminated in a specific combination to represent numeric digits or characters. To manage the clocking, a stable 50 MHz or higher clock source is used, which is then divided by the FPGA to create the necessary slower clock signals for the operation of the system. Additionally, a reliable power supply, usually 5V or 3.3V depending on the FPGA model, is used to power the FPGA and the displays.

After the hardware is set up, the next step is to implement the software that will run on the FPGA. The Verilog code for the system is the core component of the real-time implementation. The code is responsible for

converting binary-coded decimal (BCD) inputs into the corresponding display patterns for the seven-segment display. It also manages the timing of the display updates, ensuring that the system operates smoothly. The bcd7seg module takes a 4-bit BCD input and converts it into a seven-bit output that drives the segments of the display. The slow_clock_1hz and slow_clock_100hz modules generate clock signals for timing the scrolling of the message and the multiplexing of the displays. The multiplexing is essential for controlling which display is active at any given time. The scrolling_message module is responsible for managing the display's data, shifting it to create a scrolling effect on the seven-segment displays. It ensures that the message is continuously updated and that each display is properly refreshed.

Once the Verilog code is written, the next step is to test the system. Testing is a critical aspect of real-time implementation, as it ensures that the hardware and software work together as expected. The clock signals generated by the FPGA are tested to ensure they are running at the correct frequencies. The BCD-to-seven-segment conversion is also tested by checking that the correct segments light up for each possible BCD input. The multiplexing logic is tested to verify that only one display is active at any given time, with the system rapidly switching between the displays to create the appearance of simultaneous activity. The scrolling logic is checked to ensure that the message moves smoothly across the display, updating every second. During testing, any issues with timing or synchronization are addressed to prevent flickering or jerky movement in the scrolling message.

After the initial tests, further adjustments are made to optimize the system's performance. Clock frequencies may be tweaked to ensure the scrolling message is displayed at the desired speed, and power consumption may be analyzed to ensure that the system is operating efficiently. Timing delays between updates are adjusted to prevent visible flicker and to ensure that the display updates smoothly. The FPGA's ability to synchronize the clock signals with the display outputs is critical to maintaining the visual integrity of the scrolling message.

Once all issues are resolved, the system is verified and ready for deployment. The final setup involves loading the Verilog code onto the FPGA, connecting the hardware components, and observing the real-time behavior of the display. The FPGA continuously generates the necessary control signals for the seven-segment displays, updating the message in real-time and ensuring that the multiplexing and timing are handled efficiently. The final system can be used in various applications where real-time display of information is required, such as clocks, counters, or message boards. This system provides a dynamic and efficient solution for displaying information on seven-segment displays in real-time.

In conclusion, the real-time implementation of a seven-segment display controller on an FPGA involves careful hardware setup, software development, and testing. The system relies on the FPGA to handle the logic for decoding BCD values, generating clock signals, and controlling the multiplexing of displays. Real-time performance is achieved by synchronizing the clock signals and ensuring smooth updates to the display. After thorough testing and optimization, the system provides a reliable and efficient way to display scrolling messages on seven-segment displays, making it suitable for a variety of real-time display applications.

5. Simulations

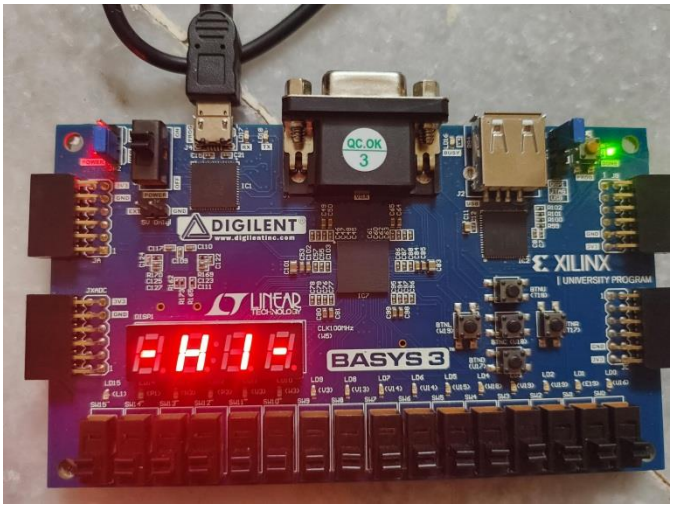

Fig -1: Result

6. ADVANTAGES

- **High Customizability:**
- FPGA allows for high flexibility in terms of design. The logic for controlling the seven-segment display and handling the scrolling message can be easily customized to meet specific requirements, such as adjusting the display speed, adding more digits, or creating complex display patterns.

- **Parallel Processing:**
- FPGAs can process multiple tasks simultaneously, enabling real-time control of multiple seven-segment displays. This parallelism ensures that the system can manage several operations without delay, providing efficient performance.

- **Low Latency:**
- FPGA-based designs generally have lower latency compared to software-based solutions because the hardware executes operations in real time. The control of the display and clocking signals happens almost instantaneously, resulting in

smoother scrolling messages and more responsive displays.

- **Scalability:**
- The FPGA system can be easily scaled to support more seven-segment displays or to handle more complex operations. If the need arises to display more digits or additional information, the FPGA logic can be modified to accommodate the change with minimal effort.

Power Efficiency:

- FPGA-based systems are generally more power-efficient than software-driven solutions that run on general-purpose processors. The hardware implementation of the seven-segment display control ensures that the system consumes less power, especially when compared to microcontroller-based systems that might require additional components for the same task.

- **Reduced System Complexity:**
- Using an FPGA to control the seven-segment displays simplifies the design and reduces the need for additional microcontrollers or external components. This can lead to a more compact and cost-effective solution.

- **High-Speed Operation:**
- FPGAs can operate at high clock speeds, enabling faster updates and smooth scrolling of messages. The ability to generate precise timing signals, like the 1Hz or 100Hz clocks, ensures that the display updates occur at the desired rate, providing smooth transitions and clear visibility.

- **Enhanced Reliability:**
- Since the logic is hardcoded into the FPGA, the system is less prone to software errors or crashes. Once programmed, the system operates predictably and consistently, offering a reliable solution for applications requiring continuous operation.

- **Real-Time Operation:**
- The FPGA-based system provides true real-time control, ensuring that the data displayed on the seven-segment displays is updated at regular intervals with no delay, which is essential for dynamic displays and real-time applications like clocks or counters.

- **Cost-Effective for Simple Applications:**
- For relatively simple applications like controlling a few seven-segment displays, using an FPGA can be more cost-effective than using a full-fledged microcontroller, as FPGAs can perform these tasks without needing additional processors or support chips.

- **Increased System Robustness:**
- Since the FPGA performs all the logic operations directly on hardware, the system becomes more robust against external interferences such as noise or signal delays. This is particularly beneficial in industrial environments or systems where high reliability is critical.

- **Ease of Debugging and Testing:**
- FPGA designs can be tested and debugged in real-time using simulation tools and on-board debugging features. This makes it easier to identify and resolve issues during development compared to software-based designs.

- **Reusability of Components:**
- The modular nature of FPGA designs allows individual components of the system, such as the BCD-to-7-segment conversion logic or clock generation modules, to be reused across different projects, reducing development time for future applications.

- **Flexibility for Future Enhancements:**
- With FPGA, future enhancements such as adding different types of displays, interfacing with other peripherals, or integrating more complex logic can be implemented without major redesigns of the system. This makes it easier to adapt the system to evolving requirements.

- **Direct Control over Hardware:**
- FPGA provides direct control over the hardware without relying on an operating system or external software. This leads to more predictable and deterministic behavior, which is ideal for real-time control systems like display drivers.

8. CONCLUSION

In conclusion, the project "Display on FPGA with 7-Segment Controller" presents an innovative and efficient approach to controlling seven-segment displays using an FPGA. The use of FPGAs for display control is highly advantageous due to their flexibility, high speed, low latency, and ability to process tasks in parallel. This project has demonstrated that FPGA-based systems can offer significant improvements in terms of customizability, power efficiency, and scalability when compared to traditional microcontroller-based approaches. Additionally, the system can be easily adapted for different applications, including dynamic displays and real-time applications, by simply modifying the FPGA logic.

One of the key benefits of using FPGA in this project is the ability to tailor the hardware to specific needs without being constrained by the limitations of microcontrollers. FPGAs allow for easy customizations, whether in terms of display size, timing control, or display patterns, which provides immense flexibility for future upgrades or adjustments. This ability to modify hardware directly, without the need for additional external components or software, simplifies the system and reduces potential points of failure, making it a highly reliable solution for real-time display control.

The parallel processing capability of FPGAs is another notable advantage. In this system, the FPGA processes multiple display segments simultaneously, ensuring that the display updates smoothly and with minimal delay. The high-speed operation of the FPGA ensures that the message scrolling, as well as the control of the seven-segment displays, occurs without lag. The real-time operation of the system, facilitated by FPGA's hardware nature, guarantees that the data presented on the display is constantly updated with precision, which is especially important in dynamic display systems like clocks, counters, or real-time message boards.

REFERENCES

[1] Xilinx Inc. (2021). FPGA Overview and Applications. Retrieved from Xilinx.com
This reference provides a comprehensive overview of FPGAs, their applications, and how they are used in various embedded systems, including display control.

[2] Mileham, D. (2014). FPGA-Based Display Systems: A Practical Approach. Wiley Series in Embedded Systems, 2014. ISBN: 978-1119994739.

A book that explores practical FPGA-based display systems and provides insight into the hardware and software development aspects for controlling displays such as seven-segment displays.

[3] Wakerly, J. F. (2006). Digital Design: Principles and Practices. 4th ed., Pearson Prentice Hall.
This book covers digital systems design, including FPGA programming, and provides the fundamental principles for designing with hardware description languages like Verilog and VHDL.

[4] Haskell, D. R., & Kuo, S. M. (2013). Digital Signal Processing: A Practical Guide for Engineers and Scientists. 2nd ed., Elsevier.
A practical guide to digital signal processing, which can be helpful in understanding how to work with FPGA-based signal processing systems like the one in your project.

[5] Brown, S. & Vranesic, Z. (2009). Fundamentals of Digital Logic with VHDL Design. McGraw-Hill.
This text introduces the fundamentals of digital logic and provides practical examples of VHDL design, which is crucial for FPGA-based system design.

[6] J. Bhasker. (1999). Verilog HDL: A Guide to Digital Design and Synthesis. 2nd ed., Prentice Hall.
This book is a great resource for understanding how to write and simulate Verilog code for digital designs like the seven-segment display controller in your project.

[7] Eshraghian, K., & Lousberg, D. (2013). Digital Systems Design with FPGA: Implementation Using Verilog and VHDL. Cambridge University Press.
This text specifically focuses on FPGA design using Verilog and VHDL, with examples for creating display systems, which would be very relevant for your project.

[8] Mohan, R., & Rathi, P. (2020). "Design of a Scrolling Message Display System Using FPGA". International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, Vol. 9, Issue 5. DOI: 10.15662/IJAREEIE.2020.0905092.
This paper discusses the design of a scrolling message display system using FPGA, similar to the scrolling message feature in your project.

[9] Vahid, F., & Givargis, T. (2002). Embedded System Design: A Unified Hardware/Software Introduction. Wiley.

A foundational text for embedded systems design, covering both hardware and software integration, which is essential for understanding FPGA-based projects.

[10] Radhakrishnan, S., & Addepalli, S. (2015). FPGA-based Real-Time Signal Processing Systems. Springer.
This book explores real-time signal processing techniques using FPGAs, which could provide deeper insights into time-critical applications like your real-time scrolling display.